209.1099 #24

D. Bond
8/19/02

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Appellants:	Howard Udell et al.
Serial Number:	09/098,204
Filed:	June 16, 1998
Entitled:	**SELF-DESTRUCTING DOCUMENT AND E-MAIL MESSAGING SYSTEM**
Examiner:	Thong Vu (Art Unit 2756)

BOX: APPEAL
Assistant Commissioner for Patents
Washington, DC 20231

August 12 2002

APPELLANTS' BRIEF ON APPEAL UNDER 37 C.F.R. § 1.192

S I R:

Appellants submit this brief for the consideration of the Board of Patent Appeals and Interferences in support of their appeal of the Final Rejection dated October 31, 2001 in the above-identified application. A Notice of Appeal was filed on April 5, 2002, and an Amendment Under 37 C.F.R. § 1.116 and a Petition for a Two-Month Extension of Time are filed concurrently herewith. An original and two copies of this brief are submitted herewith. The statutory fee of $320.00 is paid concurrently herewith.

I. REAL PARTY IN INTEREST

The real party in interest is Purdue Pharma LP, a U.S. corporation having a place of business at One Stamford Forum, Stamford, CT 06901, USA, assignee of the entire right, title and interest in the above-identified patent application. The invention was assigned by the inventors Howard R. UDELL, Stuart D. BAKER, Cary S. KAPPEL, Greg M. SHERMAN and William RIES to Purdue Pharma LP in July and September 1998. The assignment was recorded on September 4, 1998 at reel 9442, frame 0838.

II. RELATED APPEALS AND INTERFERENCES

Appellants and their legal representatives and assignee are not aware of any appeal or interference that directly affects, will be directly affected by, or will have a bearing on the decision in this appeal.

III. STATUS OF THE CLAIMS

Claims 1-10, 13-15, 17-47 are pending in this application. No claims have been allowed, all claims being subject to a final rejection dated October 31, 2002, as narrowed by an Advisory Opinion dated April 19, 2002, and it is from this final rejection that this Appeal is taken. Claims 20-43 are being canceled in an Amendment Under 37 C.F.R. § 1.116 that is being filed concurrently herewith. After this amendment, claims 1-10, 13-15, 17-19 and 44-47 remain in the application and are appealed. A copy of these appealed claims is attached hereto as Appendix A.

IV. STATUS OF AMENDMENTS

In an Amendment Under 37 C.F.R. § 1.116 filed concurrently herewith, claims 20-43 have been canceled without prejudice to Appellants' rights to reintroduce these claims in a subsequent continuation application. The Examiner has not yet had an opportunity to act upon this amendment, which is being presented in order to narrow grounds and to remove issues for appeal.

V. SUMMARY OF THE INVENTION

The present invention is designed to solve a particular problem in document retention programs. A document retention policy is typically implemented by a business in order to ensure that documents generated or received by the business are retained only for a specified period of time and are then destroyed. Systems that enforce document retention policies with regard to computer files typically scan a network's files periodically and delete files that were created prior to a specified date. Because such systems cannot delete files that are stored on the hard drives of individual computers, a document retention program must also be installed on each individual computer in order to access files stored there. Yet, the system will still be unable to delete files that are stored on floppy disks or other external media or that have been transferred, such as via e-mail, to computers outside the network, such as to an employee's home computer or laptop, or to a third party.

Appellants' invention solves this problem by providing a method for creating a self-destructing document, by creating an executable module that instructs a computer to automatically delete the document to which the executable module is attached when the document, based upon a preselected expiration date, is expired, and attaching the executable module to the document. (See, page 2, line 38 – page 3, line 6; page 9, lines 10-28; page 10, line 21 – page 11, line 31; and Figures 2-4) Examples of executable modules include an executable code, an executable program or a macro, and the executable module could execute when the document is opened. The document could also be encrypted, such that the executable module instructs the computer to decrypt the document if it is not expired and to delete the document if it is expired. (See, page 3, lines 8-12; and page 18, line 31 – page 19, line 3)

Appellants also solve this problem by providing a self-destructing e-mail messaging system having an executable module that is configured to instruct a computer to automatically delete a message to which the executable module is attached when the message, based upon a preselected expiration date, is expired. The claimed system also has an e-mail messaging system that is configured to create and transmit the message and that attaches the executable module to the message prior to transmission. (See, page 3, lines 22-31; page 9, lines 10-28; page 14, line 5 – page 18, line 28; and Figures 2 and 7a-e) Examples of executable modules include an executable code, an executable program or a macro, and the executable module could overwrite the message with null characters. (See, page 9, lines 25-28) The executable module may execute when the e-mail message to which it is attached is opened and then delete the message if the message is expired. Furthermore, the message could be encrypted, such that the executable module instructs the computer to decrypt the message if it is not expired and to delete the message if it is expired. (See, page 18, line 31 – page 19, line 3)

In another embodiment of a self-destructing e-mail messaging system, the executable module instructs a computer to automatically delete an e-mail message to which the executable module is attached when a predetermined condition, selected from the group consisting of an attempt to print the message, an attempt to copy the message and an attempt to forward the message, is met. (See, page 3, lines 8-12; page 9, line 30 – page 10, line 10)

VI. ISSUES PRESENTED FOR APPEAL

The following two issues are presented for appeal:

(1) Whether claims 1-10, 13-15, 17-19 and 44-47 are unpatentable under 35 U.S.C. § 103 as being obvious over Wilfred J. Hansen, Enhancing Documents With Embedded Programs: How Ness Extends Insets in the Andrew ToolKit, IEEE, pp. 23-32 (1990) ("Hansen") in view of U.S. Patent No. 5,903,723 to Beck et al. ("Beck"); and

(2) Whether claims 1-10, 13-15, 17-19 and 44-47 are unpatentable under 35 U.S.C. § 103 as being obvious over U.S. Patent No. 6,006,328 to Drake ("Drake") in view of U.S. Patent No. 5,787,247 to Norin et al. ("Norin").

VII. GROUPING OF CLAIMS

The Examiner has rejected all of claims 1-10, 13-15 and 17-47 as a single group. However, Appellants believe that the remaining claims 1-10, 13-15, 17-19 and 44-47 on appeal may be divided into four (4) groups for appeal. As argued below, Appellants assert that these groups of claims are separately patentable, and the claims of each group stand or fall together.

Group I includes claims 1-4 and 44-47.
Group II includes claims 6-10, 14-15 and 17.
Group III includes claims 5 and 13.
Group IV includes claims 18 and 19.

VIII. ARGUMENTS

A. Summary

The claimed invention is directed to a document or e-mail message that has an attached executable module that destroys the very document or e-mail message to which it is attached. In the following arguments, Appellants contrast this invention to each of the prior art references, which, to the extent that they teach file or e-mail message deletion at all, teach destruction of the document or e-mail message only by an external program, i.e., a program that is not attached to the document or e-mail message that is to be deleted. Appellants note that deletion by an external program is one of the problems that the present invention was designed to solve. As explained in the background section of Appellants' application, deletion by an external program of a document or an e-mail requires that the external program have access to the document or e-mail. Therefore, such a system is inadequate to ensure deletion of documents or e-mail messages that have been forwarded to or saved on a non-networked computer or external media. See, e.g., page 2, lines 6-19 of Appellants' application.

B. Rejection Based Upon the Hansen-Beck Combination

The first issue presented is whether claims 1-10, 13-15, 17-19 and 44-47 are unpatentable under 35 U.S.C. § 103 as being obvious over Hansen in view of Beck. Appellants submit that the Examiner's final rejection is in error and should be reversed.

1. The Examiner's Rejection

In the final Office Action, the Examiner stated that, as per claim 1, Hansen discloses a method for creating a self-destructing document, comprising the steps of creating an executable module which instructs a computer to automatically delete the document to which the executable module is attached when the document, based on a preselected expiration date is expired; attaching the executable module to the document (Hansen at page 28, column 2, lines 4-13 is cited in support). The Examiner admitted that Hansen fails to detail when a preselected expiration date is expired but alleged that Beck discloses an e-mail message with attachment automatically deleted by a time limit and encryption and decryption keys (Beck at column 7, lines 1-18 is cited in support). According to the Examiner, it would have been obvious to combine the technique of an e-mail message automatically deleted by an expiration date as taught by Beck and Hansen's system, as doing so would improve the security and reliability for message storage and transaction between client/server.

With respect to claims 2-4, the Examiner alleged that Hansen-Beck disclose that the executable module is an executable code, program, macro as an inherent feature of software code

(Hansen at page 28, column 2, lines 4-13 is cited in support). With respect to claim 5, the Examiner alleged that Hansen-Beck disclose the step of executing the executable module when the document is opened (Hansen at page 28, column 2, lines 4-13 is cited in support).

Regarding claims 6-10, 13-15, 17-19 and 44-47, the Examiner alleged that these claims contain the same limitations that were addressed in rejecting claims 1-5. The Examiner took Official Notice that the technique of activation of self-destruction of data, messages or software whenever a user attempts to access an unauthorized feature is well-known in the network security art (U.S. Patent No. 5,410,598 to Shear and U.S. Patent No. 5,958,005 to Thorne et al. were cited in support). By the same rationale, the Examiner rejected claims 6-10, 13-15, 17-19 and 44-47.

In the Advisory Action, the Examiner responded to Appellants' arguments in the Response to Final Office Action dated April 5, 2002. In response to arguments by Appellants, as restated by the Examiner that "the prior art does not teach the e-mail message, embedded function, automatically deleting a document", the Examiner alleged that Hansen discloses:

> enhancing documents with embedded [Hansen title, page 25 col 2, page 27 col 2, page 28 col 2] multi media mail [Hansen page 23 col 2] such as birthday card embedded a visible cake [Hansen page 30 col 2] automatically delete a file such as by executing a program which cause delete a file or virus attacks [Hansen page 28 col 2].

According to the Examiner, Hansen teaches a method using an electronic message with an embedded execute program that causes a file to be deleted or a virus to attack.

With respect to claims 18-19, in response to Appellants' argument that the prior art does not teach the encryption element, the Examiner stated that the prior art taught encryption element such as C code [Hansen page 26 col 2].

The Examiner responded to Appellants' arguments that the prior art does not teach creating a script to delete a file by stating as follows:

> Examiner notes the prior art taught when I execute a program written by someone else it may do anything I myself do, in particular delete a file [Hansen page 28 col 2]. It is obvious the program such as visible cake could delete a file, a song text or the whole email as well as activate the virus in embedded object."

With respect to claims 6 and 17, the Examiner responded to Appellants' arguments that the prior art does not teach deleting script or an executable file by stating that the prior art taught that "a program when [executed may delete] a file [Hansen page 28]; a predetermined condition is selected such as the condition of a visible cake".

2. The Hansen Article

Hansen relates to the uses of enhancing documents by embedding programs in documents and the problems associated therewith. Hansen discusses the user's ability, using a programming language named Ness, to enhance digital documents with embedded functions, for example, embedding a graphics animation into a digital birthday card. According to Hansen, objects are inserted into a document at various places, and the behavior of the objects is controlled by a script written in the programming language. See Hansen, at page 23, column 2, lines 12-14.

On page 24, column 1, at line 9, Hansen describes embedding scripts by stating:

> It is important to distinguish the notion of enhancing a document with a script from the various authoring languages for educational tutorials. ... With the latter, the author constructs a program which *generates* a sequence of images; this contrasts with the enhanced document approach of Ness where the program [sic, is] within the images. The crucial difference is in user control: with an enhanced document the reader is in control and can employ ordinary text operations to move through the text. With program generated images control lies with the system; the reader can move only to where the system allows. (emphasis in original)

Thus, in the system of enhanced documents having embedded programs described in Hansen, the reader is in control of the text, whereas, in other systems, the reader of a document cannot modify the programs that are already embedded in the document.

Hansen acknowledges that there are, however, security problems that arise with scripts that are embedded as in the Hansen-enhanced documents. As stated by Hansen, at page 28, column 2, lines 4-12, in the portion cited by the Examiner:

> Embedding of scripts in documents does not introduce a new level of security, but makes more obvious a common security problem. The problem is that in small operating systems when I execute a program written by someone else it may do anything I myself may do: in particular, delete a file, modify a file, or send a copy of a file.... Since a Ness script is a program, and since it can do anything a user can, its execution is a security loophole.

Here, Hansen acknowledges a known problem, that embedding scripts in a document serves to detract from the security of a document by ceding control to the embedded script, thereby making it less controllable by the user. Hansen's solution is to allow the user to choose whether or not to empower the embedded script. Hansen does this by providing a warning text that surrounds the Ness script and that describes the potential dangers of executing the script, such as: "Warning: Empowering a Ness script is just like running a program. The author of the script or program -- if malicious -- can write it in such a way that it can destroy your files. If you do not trust the place or person from which you got this script, DO NOT EMPOWER IT." (See Hansen, Appendix B) Only if the user decides to empower the script will the action dictated by the script be executed. However, it is important to note that Hansen considers an executable module that is embedded in a document to be part of the problem under discussion, not part of the solution.

3. U.S. Patent No. 5,903,723 (Beck et al.)

Beck relates to a system for transmitting e-mail attachment references in a network. In this system, an attachment is stored with a unique network address in a storage device visible to the network and relatively local to the sender. An attachment reference comprising the network address of the attachment is generated and is transmitted from the sender to the recipient. Only the attachment reference, i.e., the network address, but not the attachment itself, is sent. Beck compares this system to a hypertext link to a site on the worldwide web (see Beck, at column 4, lines 1-55). One purpose of the Beck system is to use storage, processing bandwidth and processing bandwidth more efficiently, since many attachment messages are transmitted but are never viewed or used by the recipient (see Beck, at column 4, line 56 – column 5, line 5).

In the portion of Beck cited by the Examiner, at column 7, lines 1-18, Beck states:

> Alternatively, instead of the retrieval of attachment 420 being made transparent to the user of PC 212, the user may be given the option to view information about the remotely-stored attachment 420 before deciding to fetch this attachment. Additionally, as will be appreciated, an attachment may in alternative preferred embodiments be automatically deleted after being accessed by the recipient or by all of the recipients, where multiple recipients receive the e-mail message; or after a given time limit, such as 90 days. As will also be understood, one additional advantage of the present attachment by reference invention is that the locally-stored attachment may in some embodiments be

> updated if necessary, in some cases before a recipient has decided to read the attachment. Thus, although an attachment is distributed at a particular time before subsequent changes are made, at least some intended recipients may get a more up-to-date copy of the attachment when they finally decide to read the e-mail attachment. (emphasis added)

Here, Beck states only that the document (called "the attachment", although it is not actually attached to the e-mail but rather is stored elsewhere on the system) is identified by the attachment reference and may be deleted or updated, perhaps as part of a remote application. By automatic deletion, Beck is referring to the operation of a standard document retention program that deletes documents on a system after a specified time period, which is part of the actual problem discussed by Appellants in the background section of their application at page 2, lines 6-19.

It should be noted that, because the document to be viewed is never actually attached to the e-mail (as confirmed in the portion of Beck immediately preceding the quoted portion, at column 6, lines 38-67), the deletion of this document cannot be done by an executable program or module that is attached to the e-mail. In fact, the e-mail "attachment" reference pointer in Beck is not itself executable. Thus, no program or module attached to the e-mail itself deletes any data. Moreover, Beck contains no suggestion that the automatic deletion of the document is done by an executable program attached to the "attachment" document.

4. The Hansen-Beck Combination Does Not Render the Claims Obvious

Group I: Claims 1-4 and 44-47

Hansen does not disclose a method for creating a self-destructing document as claimed in claims 1-4 and 44-47, as alleged by the Examiner. Rather, Hansen discusses issues relating to embedding programs in documents and the solutions adopted in the Ness component of the Andrew ToolKit. One of these issues, as discussed at page 28, beginning at line 3 of Hansen, is the "common security problem" that embedding scripts within documents "makes more obvious". The security problem, as stated by Hansen, is that, since a Ness script is a program that can do anything a user can do, execution of a program written by someone else is a security loophole and could lead to deletion of files. Hansen's proposed solution to the known security loophole is wrapping the script in a very visible warning message, thus allowing the user of the document to circumvent (i.e., not execute) the embedded program.

This is the well-known problem of Trojan horse-type viruses: a file or virus that executes an embedded program attached within the file. In this common security problem, the embedded program can operate on or be destructive towards other files in the user's operating system. This is what is meant when Hansen says "a program written by someone else ... may do anything I myself may do: in particular, delete a file, modify a file, or send a copy of a file". Hansen thus refers only to the known problem of programs that take operations on other files within the user's operating system and the ability of the user to circumvent these programs. Hansen does not discuss or suggest an embedded program that operates on the very document to which it is attached, leaving the user no choice as to whether or not to execute it. The Examiner has misinterpreted Hansen and has misapplied what is a statement of known prior art.

By contrast, Appellants' invention in claim 1, the only independent claim in Group I, is not at all related to the issue discussed in Hansen. Appellants' invention is directed to the use of a module attached to a document for deleting the very document to which it is attached, based upon the preselected expiration date of that document. The module in Appellants' claim 1 is not accessible by the user who views or opens the document. Moreover, the user has no choice as to whether or not to execute the module attached to the document, since the module operates on its own to self-destruct the document based solely upon the preselected expiration date.

This difference between Hansen and the invention in claim 1 is important to note. Hansen discusses programs embedded in documents and the ability of the user to avoid executing those programs by providing an appropriate warning of the presence and potential destructiveness of the program. According to Hansen, "with an enhanced document the reader is in control," as evidenced by the user's ability to avoid executing embedded programs that may be destructive to other files. By contrast, in Appellants' claim 1, control over the document lies not with the user but rather with the attached executable module, which "instructs a computer to automatically delete the document to which the executable module is attached". In the claimed invention, as opposed to in Hansen, the user has no control over the execution of the program and over whether or not to delete the document to which it is attached.

According to the Examiner, Hansen discloses the steps of claim 1's method for creating a self-destructing document, except for deleting the document at a preselected expiration date. This is not correct. Hansen discusses embedding scripts into documents to enhance their appearance and ornamental effects, not attaching executable modules to documents to perform self-destruction. Contrary to the Examiner's position, Hansen does not mention or even suggest a method for creating a self-destructing document. In fact, Hansen makes no mention of creating a script or an executable module to automatically delete the document to which it is attached. To the contrary, Hansen references deletion only in the context of a program that, if executed, may delete a different file, thus teaching away from the claimed invention which requires that an executable module embedded in a document delete that very same document. Security is discussed in Hansen only as a concern regarding the damage that the script can do to a system. Hansen does not teach or suggest a security system in which automatic self-deletion of documents is implemented by any computer functionality outside control of the user, let alone an executable file for this purpose.

Claim 1 requires creation of "an executable module that instructs a computer to automatically delete the document to which the executable module is attached" and attaching the separate, executable module to a document. The executable module, which is not functional without the document to which it is attached, directs a processor to delete the document. Since the executable module is a part of the document, the instructions to delete the document come from the document itself, thereby accomplishing the claimed "self-destruct." In Hansen, there is no automatic self-deletion of a document and no attaching to a document of an executable module that deletes that attached document. Thus, Hansen does not teach these steps of claim 1.

The Examiner admitted that Hansen fails to detail that deletion occurs when a preselected expiration date is reached but alleged that Beck discloses an e-mail message with attachment automatically deleted by a time limit. However, Beck does not solve the deficiencies of Hansen with respect to claim 1. Whereas Beck may state that "an attachment may … be automatically deleted … after a given time limit", this is merely a discussion of well known document retention systems that operate to delete documents after a specified time period, which are discussed in the background section of Appellants' patent application. Beck still presents no solution to the problem stated by Appellants of how to automatically delete documents that are scheduled to be deleted by a document retention program but to which the document retention program has no access because these documents have been saved or forwarded out of the network.

It should be noted that Beck does not disclose a document having an attached module that serves to operate on the very document to which it is attached, deleting it at a certain expiration date, i.e., self-destruction. In Beck, it is the document that is referenced by the address reference that is deleted, rather than any document that is actually attached to the e-mail, or even the e-mail message itself, which remains untouched. Furthermore, there is no suggestion in Beck as to how deletion of the referenced document is done, and there is certainly no suggestion that the deletion is performed by an executable module attached to the document, as required by claim 1.

Thus, even the combination of Hansen and Beck does not render claim 1 obvious, as the combination still has no teaching or suggestion of a method for creating a self-destructing document in which documents are automatically self-deleted by a computer under control of an executable file that was created for this purpose, was attached to the document and is completely outside control of the user. Accordingly, Appellants believe that claim 1 is patentable over the Hansen-Beck combination of references and respectfully request that this rejection be reversed. Claims 4 and 44-47 depend from and include all the limitations of claim 1, and Appellants submit that the rejection of these dependent claims should also be reversed.

Group II: Claims 6-10, 14-15 and 17

Claims 6-10, 14-15 and 17 were rejected along with the claims of Group I. However, Appellants believe that these claims are separately patentable from those of Group I because these claims relate to an e-mail messaging system, whereas the claims of Group I relate to a method for creating a self-destructing document. Each of the claims of Group II requires an e-mail messaging system that is configured to create the message, transmit the message and attach the executable module to the message prior to transmission. This element is not disclosed or suggested in the prior art and constitutes a novel element in claims 6-10, 14-15 and 17 that makes these claims separately patentable from the claims in Group I. In the arguments below, Appellants direct their arguments to claims 6 and 17, the two independent claims of Group II.

Hansen does not disclose the self-destructing e-mail messaging system claimed in claims 6 and 17, as alleged by the Examiner. Hansen discusses the known security problem of programs that take operations with respect to other files within the user's operating system, and Hansen intentionally gives the user the ability to avoid executing the embedded program. Both claims 6 and 17, by contrast, are directed to a self-destructing e-mail messaging system wherein an executable module attached to an e-mail message deletes the very e-mail message to which it is attached. Since the module operates on its own, based solely upon a preselected expiration date (claim 6) or a predetermined condition such as an attempt to print, copy or forward the message (claim 17), the deletion is automatic, and the user has no choice as to whether or not to execute the module attached to the e-mail message. In addition, Hansen does not mention or even suggest a self-destructing e-mail messaging system. The Examiner has misapplied Hansen, since Hansen does not discuss or suggest an e-mail messaging system wherein an executable module operates on and self-destructs the very e-mail message to which it is attached, leaving the user no choice as to whether or not to execute the module and delete the e-mail message.

It is thus not correct for the Examiner to say that Hansen discloses the elements of the claimed self-destructing e-mail messaging system, except for deletion of the e-mail message upon a preselected expiration date or a predetermined condition. The systems of claims 6 and 17 require "an executable module" that instructs a computer to "automatically delete an e-mail message to which the executable module is attached" and an e-mail messaging system that creates the message, transmits the message and attaches the separate, executable module to the message prior to transmission. Since the executable module is now part of the e-mail message and directs a processor to delete the e-mail message, the instructions to delete the document come from the e-mail message itself, thereby accomplishing a "self-destruct." Hansen contains no reference or suggestion to automatic self-deletion of an e-mail message and no attaching to an e-mail message of an executable module that deletes that attached e-mail message. Instead, as noted above, Hansen discusses how to avoid executing a program that is able to delete a second document while being itself embedded in a first document, thus teaching away from the claimed invention. Therefore, Hansen does not teach these elements of claims 6 and 17.

The Examiner further stated in the Advisory Action that Hansen taught that a program when executed may delete a file when a predetermined condition is selected, such as the condition of a visible cake, thus supposedly finding support in Hansen for claim 17's requirement that deletion of the e-mail message occur upon meeting of a predetermined condition, such as an attempt to print, copy or forward the message. However, Hansen does not teach this. The visible cake shown in Hansen is the way in which the user empowers the "Happy Birthday" script, and the image of the cake is changed as the text of the song is posted on the screen. The candles lit or unlit condition of the visible cake is a manifestation of the "showcake" function and consists of a cake image with unlit candles being switched with another cake image with lit candles. Hansen does not state that the condition of the visible cake causes deletion of any document, and there are no predetermined conditions that must be met in Hansen before the program may be executed. Thus, there is no hint or suggestion in Hansen to use the condition of the visible cake as a predetermined condition that must be met before the program script may be executed.

The Examiner admits that Hansen fails to detail that deletion of the e-mail message occurs when a preselected expiration date is reached, as claimed in claim 6, but alleges that Beck discloses an e-mail message with attachment automatically deleted by a time limit and encryption and decryption keys. However, whereas Beck discusses well known document retention systems that operate to delete documents after a specified time period, which are discussed in the background section of Appellants' patent application, Beck still presents no solution to the problem stated by Appellants of e-mail messages that are scheduled to be deleted by a document retention program but to which the document retention program has no access because these e-mail messages have been saved or forwarded out of the network.

Moreover, Beck does not disclose an e-mail message with an attachment that is automatically deleted by a time limit, as maintained by the Examiner. Neither the e-mail message nor any program attached to it is actually deleted in Beck. Instead, only the document that is referenced by the address reference is deleted, not any actual file attached to the e-mail or the e-mail itself, which remains untouched. Beck does not disclose any executable module that operates on the very e-mail message to which it is attached, deleting it at a certain expiration date. Furthermore, there is no suggestion in Beck as to how the deletion of the referenced document is done, and there is certainly no suggestion that the deletion is performed by an executable module attached to the document or to the e-mail, as required by claims 6 and 17. Thus, even the combination of Hansen and Beck does not render claims 6 and 17 obvious.

Accordingly, Appellants believe that claims 6 and 17 are patentable over the Hansen-Beck combination of references and respectfully request that these rejections be reversed. Similarly, claims 7-10 and 14-15, which depend from and include all the limitations of claim 6, are therefore allowable for the same reasons as claim 6. Appellants submit that the rejection of these dependent claims should also be reversed.

Group III: Claims 5 and 13

Although claims 5 and 13 were rejected along with the claims of Groups I and II, respectively, Appellants believe that each of claims 5 and 13 is separately patentable from its respective group because each contains the limitation that the executable module executes when the document or e-mail message to which the module is attached is opened. This element or step is not disclosed or suggested in the prior art and constitutes a novel element in claims 5 and 13 that makes claims 5 and 13 separately patentable from the other claims in Groups I and II.

Claim 5 recites the method for creating a self-destructing document of claim 1 further comprising the step of executing the executable module when the document is opened, and the Examiner rejected this claim, stating that Hansen, at page 28, column 2, lines 4-13, also disclose the step of executing the executable module when the document is opened. Claim 13 recites the self-destructing e-mail messaging system of claim 6 wherein the executable module is configured to execute when the e-mail message to which it is attached is opened, and the Examiner rejected this claim, taking Official Notice that the technique of activation of self-destruction of data, messages or software whenever a user attempts to access an unauthorized feature is well-known in the network security art

The Examiner is incorrect with regard to the limitation of claim 5 having been disclosed in Hansen. At page 28, column 2, lines 4-12, Hansen states that "[e]mbedding of scripts in documents … makes more obvious a common security problem … that in small operating systems when I execute a program written by someone else it may do anything I myself may do".

Contrary to that stated by the Examiner, Hansen does not here disclose that the executable module is executed when the document is opened. Rather, Hansen teaches that the document may be opened without executing the executable module and that the executable module is executed only when the user intentionally executes it. This is evident by the fact that Hansen's solution to the security problem of scripts embedded in documents is to allow the user to choose whether or not to empower the embedded script by providing a warning text surrounding the script describing the potential dangers of executing the script, implying that the document may be opened and the user will then have the option of whether or not to empower the script embedded therein. It is thus clear that Hansen considers the executable modules embedded in a document to be the problem, not the document itself, and that a user may open the document without executing the script embedded therein.

Moreover, Hansen certainly does not disclose that the embedded script or executable module deletes the very document within which it is embedded, such that by opening the document the user empowers the embedded module to delete the document. Nowhere does Hansen disclose that the executable module is automatically executed when the document to which it is attached is opened.

Similarly, the Examiner is incorrect in rejecting claim 13 by Official Notice of "the technique of activation of self-destruction of data, messages or software whenever a user attempts to access an unauthorized feature". The limitation of claim 13 is that the executable module attached to the e-mail message is configured to execute and destroy the e-mail message to which it is attached when the e-mail message is opened. Whereas the Examiner may be able to cite prior art that teaches or suggests destruction of an e-mail message by an external program when a user attempts to access an unauthorized feature, the Examiner has not cited any prior art that teaches or suggests that an e-mail message automatically deletes itself, i.e., self-destruction, when a user attempts to access the e-mail message. In particular, none of the other art cited by the Examiner teaches or suggests self-destruction of a document or e-mail, and neither Hansen nor Beck discloses that an e-mail message automatically deletes itself, as a result of execution of an executable module that is attached to the e-mail message, when the e-mail message is opened. Merely asserting Official Notice does not constitute citation of prior art, and the Examiner's

Official Notice is thus not sufficient for a rejection of claim 13, since it does not relate to any limitation of claim 13. See MPEP § 2144.03.

Accordingly, the combination of Hansen and Beck, even when coupled with the Examiner's Official Notice, does not render claims 5 and 13 obvious, and Appellants respectfully request that the rejections of claims 5 and 13 be reversed.

Group IV: Claims 18 and 19

Claims 18 and 19 depend from claims 1 and 6, respectively, and further recite that the document or e-mail message is encrypted, and the executable module instructs the computer to decrypt the document or e-mail message if it is not expired and to delete the document or e-mail message if it is expired. Claims 18 and 19 were rejected along with the claims of Groups I and II, respectively. However, Appellants believe that each of claims 18 and 19 is separately patentable from its respective group because each contains this additional limitation. This limitation that the executable module instructs the computer to decrypt the document or e-mail message if it is not expired and to delete the document or e-mail message if it is expired has not been disclosed or suggested in the prior art and constitutes a novel element in claims 18 and 19 that makes claims 18 and 19 separately patentable from the other claims in Groups I and II.

With regard to claims 18 and 19, the Examiner stated that Hansen at page 26, column 2 taught encryption element such as C code. However, Hansen does not teach encryption of a document or e-mail message that is attached to an executable module. In relevant portion, Hansen states:

> The statements within an on construct may refer to *currentinset* to refer to the inset whose event has triggered the current execution; they may refer to inset (<string expression>) to refer to the inset whose name is given by the <string expression>. Both constructions yield objects as their values; objects on which it is possible to invoke two classes of function that are defined in C code. First, the script may refer to methods and instance variables of the object. Since this is not well protected and can lead to incorrect behavior, it is discouraged, but there are situations in which it is crucial. More safely, the script can call functions in what is called the *proctable.* Each inset defines a set of procedures in this table, from which they are available to be called from Ness functions or to be bound to keystrokes or menu options as a user customizes his environment. (emphasis in original)

Hansen here discusses the use of a Ness "Extend" construct, which is irrelevant to the claimed invention, although Hansen mentions that certain function may be defined in the computer code "C". However, contrary to that stated by the Examiner, nowhere in this section or anywhere else in Hansen is the use of encryption discussed. In fact, Hansen does not mention encryption anywhere in the article. Thus, the Examiner reference is misplaced and inapplicable.

Neither Hansen nor Beck discloses or even suggests the use of encryption as a security feature in a document or e-mail message that is attached to an executable module, whereby the executable module instructs the computer to decrypt the document or e-mail message to which the executable module is attached if the document or e-mail message is not expired and to delete the document or e-mail message to which the executable module is attached if the document or e-mail message is expired. The Examiner has not found the feature of claims 18 and 19 in the prior art

Thus, the combination of Hansen and Beck does not render claims 18 and 19 obvious, and Appellants respectfully request that the rejections of claims 18 and 19 be reversed.

No Prima Facie Rejection Made Based on Hansen and Beck

The Examiner, in applying Hansen and Beck, has failed even to establish a prima facie case of obviousness and to provide adequate grounds of rejection of claims 1-10, 13-15, 17-19 and 44-47, as required by MPEP § 2143.

First, the Examiner has not shown any suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to combine Hansen and Beck with respect to these claims, the first requirement to establish a prima facie case of obviousness. The motivation alleged by the Examiner for combining Hansen and Beck to develop the methods of claims 1-5, 18 and 44-47 and the systems of claims 6-10, 13-15, 17 and 19 is that the combination "would improve the security and reliability for message storage and transaction between client/server". However, as discussed below, since none of these claims is related to improving "security and reliability for message storage and transaction between client/server", the Examiner's statement of motivation is irrelevant.

Claims 1-5, 18 and 44-47 relate to a method for creating a self-destructing document, the result of which is a document that is deleted by itself upon reaching a preselected expiration date, and claims 6-10, 13-15, 17 and 19 relate to self-destructing e-mail messaging systems, the result of which is a system in which an e-mail message is deleted by itself upon reaching a preselected expiration date or upon an attempt to print, copy or forward the message. None of these claims relates to improving "security or reliability for message storage", since storage of the document or e-mail message is not a focus or goal of the claimed inventions. The document or message is stored according to standard methods, and no improvement in the security or reliability of message storage is contemplated. Quite the opposite -- the claims are directed to deletion of the document or e-mail message, not storage thereof.

Furthermore, none of these claims relates to improving "message transaction between client/server". No transactions between a client and a server are required by the invention whatsoever, and none of the claims refer to any such transaction. The transmission of the self-destructing e-mail message in claims 6-10, 13-15, 17 and 19 is done according to standard methods, and no improvement in the security or message transaction between client and server is required. Thus, because the Examiner's statement of motivation to combine the Hansen and Beck references is completely irrelevant to Appellants' claimed inventions, the Examiner has not provided any proper suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to combine Hansen and Beck with respect to the claims of the application.

The Examiner has also not satisfied the second requirement to establish a prima facie case of obviousness, i.e., showing a reasonable expectation of success that the proposed combination will work as intended by the Appellants' disclosure. First of all, the subject matters of Hansen and Beck are not at all related to each other. In relevant portions, Hansen discusses avoiding the security problem of accessing modules embedded in a document by warning users of the potential dangers of the embedded module, and Beck deals with solving the problem of limited bandwidth and storage capacity on e-mail networks by attaching to an e-mail message the network address of a document rather than the actual document itself.

One who wanted to "improve the security and reliability for message storage and transaction between client/server", as stated by the Examiner, may look to Hansen to develop a warning to help the user avoid accessing a destructive program embedded in a document, since Hansen does not otherwise deal with security issues. However, there would be no reason to look to Beck for this purpose, since Beck's system of transmitting e-mail references without the actual document attachment has no relation to security whatsoever. In fact, combining Hansen and Beck would be counter to the purposes of Beck, since adding into the e-mail of Beck a script or a warning message wrapped around a script as in Hansen would not reduce congestion on network lines, since embedding scripts in a document only adds more data to the document, thereby increasing its size and occupying more bandwidth for transmission and more storage capacity in memory. Hansen and Beck are counter-intuitive to each other and defeat each other's objectives and goals, and there is thus no logical reason to combine these unrelated references.

In addition, both Hansen and Beck actually teach away from the claimed invention. Hansen discusses embedding scripts into documents to enhance their appearance and ornamental effects and considers executable modules as security problems for the documents to which they are attached. Hansen does not discuss or suggest attaching an executable module to a document or e-mail message to perform self-destruction by the document or e-mail message as the solution to a problem, as claimed. Hansen references deletion only in the context of a program that, if executed, may delete a different file on the system, and Hansen teaches that this result is to be avoided, as evidenced by the warning suggested to be placed around the embedded program in order to protect the user, such as: "Warning: Empowering a Ness script is just like running a program. ... If you do not trust the place or person from which you got this script, DO NOT EMPOWER IT." Thus, because Hansen discusses how to avoid executing an executable module that may delete a second document while being itself embedded in a first document, Hansen teaches away from the claimed invention, which claims deleting a document by empowering an executable module embedded in that same document.

Beck also teaches away from the claimed invention. Beck relates to the problem of limited bandwidth and storage capacity on e-mail networks by attaching to an e-mail message the unique network address of a stored document rather than the actual document itself. Beck thus teaches that it is preferable not to attach actual files to an e-mail message. This is contrary the self-destructing e-mail messaging systems of claims 6-10, 13-15, 17 and 19, which all require that the executable module be attached to the e-mail message. Thus, Beck teaches away from attaching documents or other modules to e-mail messages, which is one of the required steps of the claimed inventions.

Appellants fail to see how the proposed combination of two references that are unrelated and that both teach away from the claimed inventions can have any reasonable chance of working as intended by the Appellants' disclosure. Accordingly, because Hansen and Beck are unrelated in subject matter and both teach away from the claimed invention, there is no reasonable expectation of success that combining Hansen with Beck would render obvious the methods for creating a self-destructing document of claims 1-5, 18 and 44-47 or the self-destructing e-mail messaging systems of claims 6-10, 13-15, 17 and 19.

Furthermore, the third and final requirement for a prima facie case of obviousness, the teaching or suggestion in the prior art of all the claim limitations, has also not been met for any of claims 1-10, 13-15, 17-19 and 44-47 in Groups I-IV. Neither Hansen nor Beck contains even a hint of a disclosure regarding the methods for creating a self-destructing document or the self-destructing e-mail messaging systems, as discussed separately above for the claims of each group.

C. Rejection Based Upon the Drake-Norin Combination

The second issue presented is whether claims 1-10, 13-15, 17-19 and 44-47 are unpatentable under 35 U.S.C. § 103 as being obvious over Drake in view of Norin. Appellants submit that the Examiner's final rejection is in error and should be reversed.

1. The Examiner's Rejection

In the final Office Action, the Examiner stated that, as per claim 1, Drake discloses a method for creating a self-destructing document, comprising the steps of creating an executable module which instructs a computer to automatically delete the document to which the executable module is attached when the document, based on a preselected expiration date is expired;

attaching the executable module to the document, such as a message with a header is attached by an executable code or software which is designed to self-destruct (Drake at Figure 10 and at column 7, lines 43-52 is cited in support). The Examiner admitted that Drake fails to detail when a preselected expiration date is expired but alleged that Norin discloses an e-mail message with a time-based expiration date wherein an object older than a set time will be deleted automatically (Norin at column 24, lines 1-25 is cited in support). The Examiner concludes that it would have been obvious to combine the technique of an e-mail message automatically deleted by an expiration date as taught by Norin and Drake's system, as doing so would improve the security and reliability for message storage and transaction between client/server.

With respect to claims 2-4, the Examiner alleged that Drake-Norin disclose that the executable module is an executable code, program, macro as an inherent feature of software code (Drake at Figure 10 and at column 7, lines 43-52 is cited in support). With respect to claim 5, the Examiner alleged that Drake-Norin disclose the step of executing the executable module when the document is opened (Drake at Figure 10 and at column 7, lines 43-52 is cited in support).

With respect to claims 6-10, 13-15, 17-19 and 44-47, the Examiner alleged that these claims contain the same limitations that were addressed in rejecting claims 1-5. The Examiner took Official Notice that the technique of activation of self-destruction of data, messages or software whenever a user attempts to access an unauthorized feature is well-known in the network security art (U.S. Patent No. 5,410,598 to Shear and U.S. Patent No. 5,958,005 to Thorne et al. were cited in support). By the same rationale, the Examiner rejected claims 6-10, 13-15 and 17-47.

In the Advisory Action, the Examiner responded to Appellants' arguments in their Response to Final Office Action and stated, with respect to claims 1-10, 13-15, 17-19 and 44-47, that the prior art taught self-destruction code/programs such as viruses and Trojan horses.

2. <u>U.S. Patent No. 6,006,328 (Drake)</u>

Drake relates to a computer software protection and security system, wherein certain attacks on executable software are prevented by, among other things, replacing the executable software with new versions having enhanced security. According to Drake, in the standard scenario for "running" a given executable program under the control of a computer operating system, the executable program is modified to ensure its integrity and improve its security by preventing rogue operations such as eavesdropping, disassembly and examination, tampering and execution-tracing. Thus, the main way in which Drake implements its security features is by replacing the executable software with a more secure version.

Figures 7 and 9 of Drake show a prior art format for storing executables, having (1) a header section that stores standard information required by the computer operating system for running the executable, (2) a code section for storing the "algorithmic" portion of the code, and (3) a data section for storing the data, such as constants, or overlays that are utilized by the code section. Drake describes in Figure 6 and at column 14, lines 7-32 that an executable program is transformed into a new executable by an obfuscating step that modifies the header of the executable and inserts loading code, a cipher step that encrypts the existing executable and calculates check data for the encrypted executable, and an anti-key press and authentication step that replaces various insecure system calls with safe equivalent code and may insert code to graphically represent the integrity of the executable program. The new executables are then stored on disk in place of the old executable programs.

Drake states that, in an attempt to breach security, a rogue may trace the execution of software (called debugging) in order to compromise its security. Hampering tracing prevents this, and there are numerous methods of detecting when debugging or tracing is taking place so as to make the rogue's task of detecting and bypassing debug-detection extremely difficult. In this regard, Drake states at column 7, lines 43-52, in a section cited by the Examiner, that:

> Using strong cryptographic schemes ... will present [sic, prevent] the examination of any decryption routines from revealing a simple patch to disable said routines. When tracing software, the program stack is usually used by the debugger either during the tracing operations or at other times. This is easily detected, and <u>by using the area of the stack which will be destroyed by unexpected stack-use for code or critical data, software can be designed to self-destruct in this situation</u>. (emphasis added)

This deletion function, the only deletion discussed in Drake, is a method Drake uses to hamper debugging. By intentionally storing code or critical data in a particular area of memory whose contents are deleted upon an unexpected use of memory, Drake thwarts the rogue's attempt to access the code or data through debugging, because the rogue actually deletes the very code or data he desires to access and is thereby prevented from accessing it. Drake also advises that use of strong encryption will also hamper the disabling of such prevention routines.

It should be noted, however, that Drake does not teach self-deletion of a document or information. This deletion is performed not by an executable module that is attached to the document or information but instead by tracing software, an external program.

3. U.S. Patent No. 5,787,247 (Norin et al.)

Norin involves a data replication system wherein a number of copies of certain data are maintained simultaneously on network servers and wherein each server keeps track of changes made locally to data and periodically broadcasts those changes to other servers to update their copies. Norin discloses a way to ensure that changes made to one data set are not lost by being inadvertently deleted without having been first stored in other copies of the data set. This is done by use of server communications to keep track of changes in data sets and the location of those changes to verify that changes made to a local copy of a data reside on at least one other system in the network (called a "handshake process").

Norin also considers the question of whether a node that once held changes but has already deleted them should respond to a request for verification that those changes still exist in the network. This question arises in the context of data that has been deleted automatically by time expiration. As stated at column 24, lines 1-25, the portion cited by the Examiner,

> For certain implementations of replication processing block 26 of FIG. 2, special consideration must be given to the handshaking process. For example, certain replication processes implement the concept of time-based expiration of data ... [which] refers to deleting data that is older than a specified time. One area where time-based expiration of data may be useful is in the area of E-mail messages. ... When a given data object is older than a set time, as for example two weeks, the data object is deleted automatically. In a replication environment, time-based expiration results in a situation where changes older than a certain time are deleted from a replica node. Because an enterprise comprises many different replica nodes, each replica node may expire data at a different time. ... In such an enterprise, if the replica node which never expired data wanted to delete its local copy of a data set, it would send a replica delete pending packet to the other replica nodes in the enterprise. (emphasis added)

In the context of data that has been deleted automatically under standard time-based expiration, Norin is concerned with keeping track of data that has been deleted.

It should be noted that Norin is not concerned with the actual deletion itself but rather only with its impact on a replication process. Norin's discussion of time-based expiration is merely of a standard document retention program that deletes documents or e-mails on a system after a preset time period, as discussed in the background section of Appellants' application. In fact, Norin presents no solution to the problem solved by Appellants of documents or e-mail messages that are to be deleted by a document retention program because they have expired but to which the document retention program has no access because these documents or e-mail messages have been saved or forwarded out of the network.

4. The Drake-Norin Combination Does Not Render the Claims Obvious

Group I: Claims 1-4 and 44-47

Claims 1-4 and 44-47 are patentable over the cited prior art because Drake does not disclose a method for creating a self-destructing document as claimed in claim 1, as alleged by the Examiner. Instead, Drake discusses a security system for computer software to prevent certain attacks on executable software by persons or other software. Drake implements its security features by replacing executable software with versions having increased security. Figures 7 and 9 of Drake show a prior art format for executables, including a header section that stores standard information required by the computer operating system for running the executable, as well as a code section and a data section. At columns 14-16, Drake describes that the executable program is transformed into a new executable by modifying and replacing the header and other parts of the executable. The new executable, whose format is shown in Figure 10, can then be stored on disk, replacing the old executable program.

The Examiner cites Figure 10 and column 7, lines 43-52 of Drake as disclosing a message with a header attached by an executable code or software that is designed to self-destruct. Drake discloses no such thing. In the cited portion, Drake discusses a self-protection security mechanism that destroys software by storing it in a specific part of memory (the program stack) that is erased when a software attack is detected. Drake states that "the program stack is usually used by the debugger … during the tracing operations" by the rogue, and Drake intentionally stores specific software information in the program stack so that, when the debugger is activated by a rogue to do a trace, the specific software information will be deleted. This deletion, which is the only deletion discussed in Drake, is concerned with thwarting a rogue attack and is executed by the debugger, not by the very data being deleted.

Thus, contrary to the statements by the Examiner, Drake does not show or contemplate a self-destruct function, code or program. The software or data in Drake does not destroy itself, but rather is destroyed by the debugger, which is a part of an external program trying to sabotage the protected software. No data in Drake self-destructs under control of a module embedded in the data, since no instructions originating from the data itself direct a processor to execute a deletion process. The Drake reference discloses no executable program embedded or attached in any document that causes that document to self-destruct.

The Examiner has thus interpreted Drake incorrectly. Drake does discuss the well-known problem of a Trojan horse-type virus, which executes an embedded program attached within a message to operate on or be destructive towards other files in the user's operating system, but only in the context of what a rogue might attempt to do and how Drake's system avoids such a virus. Contrary to the Examiner's statements, Drake does not disclose or suggest a method for creating a self-destructing document wherein an embedded program operates on the very document to which it is attached. The new executable code is not embedded or attached to anything in Drake, since adding security features and enhancing or replacing pre-existing executable application files is not embedding or attaching application files.

In addition, Drake does not suggest, as alleged by the Examiner, that the header of the executable can be used in any way for self-destruction. Figure 10 shows the format of the new executable to replace the old executable program, both of which contain a header section that stores standard information required by the computer operating system for running the executable. The Examiner seemingly seizes on the word "header" and disregards the fact that the header section of the executable is irrelevant to deletion of a document in Drake.

The Examiner admitted that Drake fails to detail that deletion of a document occurs when a preselected expiration date is reached but alleged that Norin discloses an e-mail message with a time-based expiration date wherein an object older than a set time is deleted automatically. To the contrary, Norin does not solve the deficiencies of Drake. Whereas Norin does mention that a data object or message may be deleted automatically after a set time, Norin does not teach a new method for doing so. It should be noted that Norin merely references an external prior art document retention program that deletes documents after a certain period of time. Norin itself is concerned with simultaneously maintaining a number of copies of certain data on network servers and keeping track of changes or deletions in the data. Norin is not concerned with the actual deletion itself but merely touches upon the prior art deletion mechanism and how it should be handled by the replication process, such that the act of deleting time-expired data is discussed only as it would impact the handshake process between the replication process and the deletion mechanism. Norin never teaches or suggests a mechanism for deletion of a document by its own self and does not disclose a self-destructing document having an attached module that serves to operate on the very document to which it is attached, deleting it at a certain expiration date.

Not only does Norin not teach or suggest that time-deletion of documents is done by the very documents to be deleted or that the deletion is performed by an executable module attached to the documents to be deleted, as required by claim 1, but Norin teaches away from the invention of claim 1. Norin keeps track of changed data in order to prevent inadvertent deletion, whereas the claims relate to an executable module that automatically and intentionally deletes the very document to which it is attached, when expired, regardless of its changes or location and without knowledge of the user of the document. Therefore, since Norin does not teach the deletion of time-expired data by itself but in fact teaches away from this concept, the scope of Norin's contribution is no more than a discussion of prior art time-deletion of documents, which is already well-known and discussed in Appellants' specification at page 2.

Accordingly, Appellants believe that claim 1 is patentable over the Drake-Norin combination of references and respectfully request that this rejection be reversed. Claims 2-4 and 44-47 depend from and include all the limitations of claim 1, and Appellants submit that the rejection of these dependent claims should also be reversed.

Group II: Claims 6-10, 14-15 and 17

As stated above, although claims 6-10, 14-15 and 17 were rejected along with the claims of Group I, these claims are separately patentable from those of Group I because they relate to an e-mail messaging system, whereas the claims of Group I relate to a method for creating a self-destructing document. Each of the claims of Group II requires an e-mail messaging system that is configured to create the message, transmit the message and attach the executable module to the message prior to transmission. This element is not disclosed or suggested in the prior art and constitutes a novel element in claims 6-10, 14-15 and 17 that makes these claims separately patentable from the claims in Group I. In the arguments below, Appellants direct their arguments to claims 6 and 17, the two independent claims of Group II.

Drake does not disclose or suggest a self-destructing e-mail messaging system as claimed in claims 6 and 17, as alleged by the Examiner. As discussed above, Drake discusses a security system for preventing certain attacks on executable software by replacing executable software with versions having increased security. Drake also discusses the use of a self-protection security mechanism that stores software in a specific part of memory that is erased when a software attack by a rogue is detected, thereby thwarting a rogue's attack. By contrast, Appellants' invention in claims 6 and 17 is directed to a self-destructing e-mail messaging system wherein an executable module instructs a computer to delete <u>the very e-mail message</u> to which the module is attached.

In the claimed invention, as opposed to in Drake, the execution of the program and deletion the e-mail message is done automatically by the executable module attached to the very e-mail message that is to be deleted, based solely upon the preselected expiration date (claim 6) or predetermined condition (claim 17), without regard to whether a rogue attack is detected. It is thus not correct for the Examiner to say that Drake discloses the steps of the claimed self-destructing e-mail messaging system, except for deletion of the document upon expiration of a preselected expiration date or upon meeting of a predetermined condition. Drake contains no reference to automatic <u>self-deletion</u> of an e-mail message and no attaching to an e-mail message of an executable module that deletes that attached e-mail message. Although Drake discusses the deletion of software or data when a debugger is used as is a part of a rogue program to sabotage protected software, the data does not <u>self-destruct</u> under control of a module embedded in that very same data but rather is deleted by an external program, i.e., the debugger.

Thus, Drake does not, as alleged by the Examiner, teach self-destruction code or programs. Executable code is not embedded or attached in Drake but rather is enhanced or replaced with modified executable code. As noted above, Drake discusses a Trojan horse-type virus only in the context of what a rogue might attempt to do and how to avoid such a virus. In addition, the header section in the executable format, shown in Figure 10, is for storing information and is modified and replaced when a new executable is stored. The header is not discussed or referenced with regard to self-destruction of a document. Contrary to the Examiner's statements, nowhere does Drake disclose any self-destructing e-mail messaging system wherein an executable module operates on the e-mail message to which it is attached.

The Examiner admits that Drake fails to detail that deletion of an e-mail message occurs when a preselected expiration date is reached, as claimed in claim 6, but the Examiner contends that Norin discloses an e-mail message with a time-based expiration date wherein an object older than a set time will be deleted automatically. However, Norin does not disclose this. Norin is concerned with keeping track of data that is changed or deleted and discusses a prior art deletion mechanism only as to how it would impact the replication process. Norin never teaches or suggests a mechanism for deletion of an e-mail message by its own self, either based upon a preselected expiration date or a predetermined condition, and does not disclose or suggest a self-destructing e-mail messaging system having an executable module that serves to operate on the very e-mail message to which it is attached. In addition, Norin, which deals with preventing inadvertent deletion, teaches away from the inventions of claims 6 and 17, which require automatically and intentionally deleting e-mail messages, when expired, even without knowledge of the users of the e-mail messages, as argued above regarding claim 1.

Moreover, both Drake and Norin discuss document systems. Neither Drake nor Norin teaches an e-mail messaging system, which is an element of claims 6 and 17. In particular, neither reference teaches an e-mail messaging system that is configured to create the message, transmit the message and attach the executable module to the message prior to transmission, as required by claims 6 and 17. Accordingly, because this element of the claims has not been taught or rendered obvious by the prior art, the rejections cannot stand.

For all of the reasons set forth above, Appellants believe that claims 6 and 17 are patentable over the Drake-Norin combination of references and respectfully request that this rejection be reversed. Similarly, the rejections of claims 7-10 and 14-15, which depend from and include all the limitations of claim 6, should also be reversed for the same reasons.

Group III: Claims 5 and 13

Although claims 5 and 13 were rejected along with the claims of Groups I and II, respectively, Appellants believe that each of claims 5 and 13 is separately patentable from its respective group because each contains the limitation that the executable module executes when the document or e-mail message to which the module is attached is opened. This element or step is not disclosed or suggested in the prior art and constitutes a novel element in claims 5 and 13 that makes claims 5 and 13 separately patentable from the other claims in Groups I and II.

The Examiner rejected claim 5, which recites the method for creating a self-destructing document of claim 1 plus the step of executing the executable module when the document is opened, stating that Drake-Norin, at Figure 10 and at column 7, lines 43-52, also disclose the step of executing the executable module when the document is opened. The Examiner also rejected claim 13, which recites the self-destructing e-mail messaging system of claim 6 wherein the executable module is configured to execute when the e-mail message to which it is attached is opened, taking Official Notice that the technique of activation of self-destruction of data, messages or software whenever a user attempts to access an unauthorized feature is well-known in the network security art

The Examiner is incorrect with regard to the limitation of claim 5 having been disclosed in Drake. At column 7, lines 43-52, Drake states that "When tracing software, the program stack is usually used by the debugger either during the tracing operations or at other times. This is easily detected, and by using the area of the stack which will be destroyed by unexpected stack-use for code or critical data, software can be designed to self-destruct in this situation". Contrary to that stated by the Examiner, Drake does not disclose that the executable module is executed when the document is opened. Rather, Drake teaches that the code or other data that is stored in the program stack is deleted by use of the stack during debugging, without the code or data having ever been accessed or opened at that location by the debugger.

In addition, Figure 10 of Drake merely shows that the format of the new executable to replace the old executable program has a header section that stores standard information required by the computer operating system for running the executable. Nowhere in Drake is it even suggested that this header section may be used to automatically execute the executable module when a document to which it is attached is opened.

Similarly, the Examiner is incorrect in rejecting claim 13 by Official Notice of "the technique of activation of self-destruction of data, messages or software whenever a user attempts to access an unauthorized feature". The limitation of claim 13 is that the executable module attached to the e-mail message is configured to execute and destroy the e-mail message to which it is attached when the e-mail message is opened. Whereas the Examiner may cite prior art that teaches or suggests destruction of an e-mail message <u>by an external program</u> when a user attempts to access an unauthorized feature, the Examiner has not cited any prior art that teaches or suggests that an e-mail message automatically deletes itself, i.e., <u>self-destruction</u>, when a user attempts to access the e-mail message. In particular, none of the other art cited by the Examiner teaches or suggests <u>self-destruction</u> of a document or e-mail, and neither Drake nor Norin discloses that an e-mail message <u>automatically deletes itself</u>, as a result of execution of an executable module that is attached to the e-mail message, when the e-mail message is opened. Merely asserting Official Notice does not constitute citation of prior art, and the Examiner's Official Notice is thus not sufficient for a rejection of claim 13, since it does not relate to any limitation of claim 13. See MPEP § 2144.03.

Accordingly, the combination of Hansen and Beck, even when coupled with the Examiner's Official Notice, does not render claims 5 and 13 obvious, and Appellants respectfully request that the rejections of claims 5 and 13 be reversed.

Group IV: Claims 18 and 19

Claims 18 and 19 depend from claims 1 and 6, respectively, and further recite that the document or e-mail message is encrypted, and the executable module instructs the computer to decrypt the document or e-mail message if it is not expired and to delete the document or e-mail message if it is expired. Although claims 18 and 19 were rejected along with the claims of Groups I and II, respectively, Appellants believe that each of claims 18 and 19 is separately patentable from its respective group because each contains this additional limitation. This limitation that the executable module instructs the computer to decrypt the document or e-mail message if it is not expired and to delete the document or e-mail message if it is expired has not been disclosed or suggested in the prior art and constitutes a novel element in claims 18 and 19 that makes claims 18 and 19 separately patentable from the other claims in Groups I and II.

The Examiner did not specifically state that either Drake or Norin taught encryption of a document or e-mail message that is attached to an executable module and that is decrypted upon opening the attachment if the document or e-mail message is not expired, as in claims 18 and 19. In fact, neither Drake nor Norin does disclose or even suggest the use of encryption as a security feature in a document or e-mail message that is attached to an executable module. Of the cited references, only Drake uses encryption but only in the context of using strong encryption so as to prevent examination of decryption routines to reveal a patch to disable those decryption routines. Certainly neither reference discloses or even suggests that an executable module instructs a computer to decrypt a document or e-mail message to which the executable module is attached if the document or e-mail message is not expired and instructs the computer to delete the document or e-mail message to which the executable module is attached if the document or e-mail message is expired. The Examiner has not found the feature of claims 18 and 19 in the prior art.

Thus, the combination of Drake and Norin does not render claims 18 and 19 obvious, and Appellants respectfully request that the rejections of claims 18 and 19 be reversed.

No Prima Facie Rejection Made Based on Drake and Norin

The Examiner, in applying Drake and Norin, has failed to even establish a prima facie case of obviousness and to provide adequate grounds of rejection of claims 1-10, 13-15, 17-19 and 44-47, as required by MPEP § 2143.

First, the Examiner has not shown any suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to combine Drake and Norin with respect to these claims, the first requirement to establish a prima facie case of obviousness. The motivation alleged by the Examiner for combining Drake and Norin to develop the methods of claims 1-5, 18 and 44-47 and the systems of claims 6-10, 13-15, 17 and 19 is that the combination "would improve the security and reliability for message storage and transaction between client/server".

However, as discussed above with respect to combination of Hansen and Beck, none of the claims relates to improving "security and reliability for message storage and transaction between client/server". Storage of a document or an e-mail message is not a focus or goal of the claimed inventions; rather, the claims are directed to deletion of the document or e-mail message from storage. Also, no transactions between a client and a server are contemplated by the invention, and the transmission of the self-destructing e-mail message is done according to standard methods. Thus, the Examiner's statement of motivation to combine Drake and Norin is irrelevant to the claims on appeal, and the Examiner has not provided any proper suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to combine Drake and Norin with respect to the claims of the application.

The Examiner has also not satisfied the second requirement to establish a prima facie case of obviousness, that there be a reasonable expectation of success that the proposed combination will work as intended by the Appellants' disclosure. First of all, the subject matters of Drake and Norin are not related. Drake is concerned with protecting the integrity of software system files and discusses a security system to prevent certain attacks on executable computer software by persons or other software. By contrast, Norin is concerned with data loss by mistake in a data replication system and discusses a system for ensuring that changed data is not lost by being inadvertently deleted (or time-deleted) without having been first stored somewhere else. One who wanted to "improve the security and reliability for message storage and transaction between client/server", as stated by the Examiner, may look to Drake for ways to enhance security of executable software code to help prevent destructiveness by rogues. However, one would have no reason to look to Norin for this purpose, as Norin's data replication system is not related to security and provides no disclosure for preventing destructiveness by rogues.

Moreover, Appellants do not see how a security-enhanced executable replacement system, as in Drake, when combined with a server communication network for keeping track of changed data with the aim of preventing inadvertent deletion, as in Norin, can be reasonably expected to produce a method of creating a self-destructing document that is automatically deleted by an attached executable module upon that document's preset expiration date, as in claim 1, or a self-destructing e-mail messaging system having an executable module configured to instruct a computer to automatically delete an e-mail message to which the module is attached when the message is time-expired or when a predetermined condition is met, such as an attempt to print, copy or forward the message, as in claims 6 and 17. Combining the systems of Drake and Norin would produce a system that has both a security-enhanced executable replacement feature, as in Drake, and a server communication network for keeping track of changed data with the aim of preventing inadvertent deletion, as in Norin. Such a system, however, would still not produce a method for creating a self-destructing document or a self-destructing e-mail messaging system as claimed.

In addition, Norin actually teaches away from the claimed invention, since the very purpose of Norin is counter-intuitive to the objectives of Appellants' claimed invention. Norin keeps track of changed data in order to prevent inadvertent deletion, whereas the claims relate to an executable module that automatically and intentionally deletes the very document or e-mail message to which it is attached, when expired, regardless of its changes or location and without knowledge of the user of the document or e-mail message.

The Examiner has not shown how these two references, which do not teach or suggest creating an executable file, attaching the executable file to a document or to an e-mail message, or automatically deleting an expired document to which an executable file is attached, can have any chance of producing a method of creating self-destructing document with those steps. Appellants fail to see how the proposed combination of two references that are unrelated and that both teach away from the claimed inventions can have any reasonable chance of working as intended by the Appellants' disclosure. There is thus no reasonable expectation of success that combining Drake and Norin would render obvious the methods for creating a self-destructing document of claims 1-5, 18 and 44-47, or the self-destructing e-mail messaging systems of claims 6-10, 13-15, 17 and 19.

Furthermore, the third and final requirement for a prima facie case of obviousness, the teaching or suggestion in the prior art of all the claim limitations, has also not been met for any of claims 1-10, 13-15, 17-19 and 44-47 in Groups I-IV. Neither Drake nor Norin contains even a hint of a disclosure regarding the methods for creating a self-destructing document or the self-destructing e-mail messaging systems, as discussed separately above for the claims of each group.

IX. CONCLUSION

Appellants' claimed methods and systems are substantially different from anything cited in the prior art references, Hansen, Beck, Drake and Norin. The claimed methods and systems have steps and elements that are not disclosed or even suggested in the cited prior art, and Appellants believe that for the foregoing reasons the final rejections of claims 1-10, 13-15, 17-19 and 44-47 must be reversed.

Prompt consideration of the arguments presented herein and reversal of the final rejections is earnestly solicited.

Respectfully submitted,
DAVIDSON, DAVIDSON & KAPPEL, LLC

By: ____________________
Morey B. Wildes
Reg. No. 36,968

DAVIDSON, DAVIDSON & KAPPEL, LLC
485 Seventh Avenue, 14th Floor
New York, NY 10018
Tel: (212) 736-1940

APPENDIX A

PENDING CLAIMS 1-10, 13-15, 17-19 and 44-47 OF U.S. PATENT APPLICATION NO. 09/098,204

1. A method for creating a self-destructing document, comprising the steps of:

creating an executable module which instructs a computer to automatically delete the document to which the executable module is attached when the document, based upon a preselected expiration date, is expired;

attaching the executable module to the document.

2. The method according to claim 1, wherein the executable module is an executable code.

3. The method according to claim 1, wherein the executable module is an executable program.

4. The method according to claim 1, wherein the executable module is a macro.

5. The method according to claim 1, further comprising the step of executing the executable module when the document is opened.

6. A self-destructing e-mail messaging system, comprising:

an executable module, the executable module configured to instruct a computer to automatically delete a message to which the executable module is attached when the message, based upon a preselected expiration date, is expired;

an e-mail messaging system, the e-mail messaging system configured to create the message and to transmit the message, the e-mail messaging system attaching the executable module to the message prior to transmission.

7. The system according to claim 6, wherein the executable module is an executable code.

8. The system according to claim 6, wherein the executable module is an executable program.

9. The system according to claim 6, wherein the executable module is a macro.

10. The system according to claim 6, wherein the executable module is configured to overwrite the message with null characters.

13. The system according to claim 6, wherein the executable module is configured to execute when the e-mail message to which it is attached is opened.

14. The system according to claim 6, wherein the executable module is configured to begin execution when the message to which it is attached is opened, the executable module deleting the message during said execution if the message is expired.

15. The system according to claim 6, wherein the e-mail message is an e-mail message attachment.

17. A self-destructing e-mail messaging system, comprising:

an executable module, the executable module configured to instruct a computer to automatically delete an e-mail message to which the executable module is attached when a predetermined condition is met, wherein said predetermined condition is selected from the group consisting of an attempt to print the message, an attempt to copy the message and an attempt to forward the message;

an e-mail messaging system, the e-mail messaging system configured to create the message and to transmit the message, the e-mail messaging system attaching the executable module to the message prior to transmission.

18. The method of claim 1, wherein the document is an encrypted document, and wherein the executable module is configured to instruct the computer to decrypt the document if the document is not expired, and to delete the document if the document is expired.

19. The method of claim 6, wherein the message is an encrypted message, and wherein the executable module is configured to instruct the computer to decrypt the message if the message is not expired, and to delete the message if the message is expired.

44. The method of claim 1, wherein the executable module overwrites the document with null characters.

45. The method of claim 1, wherein the creating step includes selecting the preselected expiration date by selecting a number of days until expiration of the document.

46. The method of claim 1, wherein the creating step includes selecting the preselected expiration date by selecting a month, date, and year.

47. The method of claim 46, wherein the month, date, and year are selected by user input.